REMARKS

In the Office Action mailed December 18, 2006, claims 1-14 and 21-27 were rejected under 35 U.S.C. §103(a) as being obvious over Bengston (U.S. Pat. No. 6,728,947) in view of Nichols et al. (U.S. Pat. No. 6,018,730) in further view of Bacon et al. (U.S. Pat. No. 6,430,538).[1]

A discussion of the disclosed embodiments of the present invention is provided below, along with a discussion of the rejections under §103(a) beginning at page 19 below.

Disclosed Embodiments

In order to provide a better understanding of the present invention, the following discussion of disclosed embodiments of the present invention is provided. In general, the present invention provides a system and method for processing workflows in real time to generate programmatic process-based decisions based on dynamically executed checklists or task lists. The system utilizes a workflow engine that is defined at design time, prior to runtime. Thus, it can be recognized that the execution of a particular process-based workflow will occur dynamically. Despite the fact that the workflow engine is defined prior to runtime, the particular order of workflow processing according to the checklist for the selected process decision is determined during workflow execution, rather than in a fixed, pre-determined sequence. Where an administrative utility or workflow designer is provided according to the present invention, associations and rules governing workflow execution are determined by an administrator or manager, but the specific functions or tasks in a checklist or task list and the actual sequence of task processing are determined during actual processing of the loan application. Particular features and components of the system 10 are described in greater detail below with respect to a disclosed embodiments that relate to loan processing decisioning.

The workflow engine 20 (a/k/a rules engine) is configured to handle complex lending rules like entry condition rules, eligibility rules, lending policies & criteria, pricing rules &

[1]On page 2 of the Office Action it is stated that "Claims 1-4 and 21-27 are rejected under 35 USC 103(a) . . . ." It is believed that this is a typographical error, and should indicate claims 1-14 and 21-27 to correspond to the substantive rejections detailed on pages 2-20 of the Office Action.

adjustments, credit scoring rules, partner product pricing criteria, etc. (See, e.g., specification pp. 7-14). The workflow engine 20 permits management of all such rules in real-time so that logical formulas are applied to workflows automatically. As explained below, rules can be modified, added, tested in simulation runs, and then published to any given system environment (e.g., a test environment or a consumer-accessible production environment).

The workflow designer 24 is browser-based and includes several functions or components: a designer component; a loan director; an e-loan director; and a bank workflow setup. (See, e.g., specification pp. 7-14). The workflow designer 24 is driven by the workflow engine 20 that operates through a program that processes individual instances of the workflow tasks "running" on a server and allows queues to be automatically or manually added as part of the load balancing options and high scalability for high-volume automated loan processing. The workflow designer 24 incorporates a rules engine that enables participating financial institutions (and business users within those financial institutions) to configure and manage "business rules" for offered loan products in real-time.

The designer component of the workflow designer 24 is used to establish tasks (a/k/a workflow processes/sub-processes/instances), checklists (a/k/a business process maps) with automation features, rules and selection criteria. These tasks are defined prior to runtime setup, although the designer component allows establishment of options or parameters associated with those defined tasks. (See, e.g., specification, pp. 19-20). In this way, the designer component allows creating a highly flexible and automated workflow, providing many options for each task within a loan processing checklist to perform specific or multiple functions, thereby allowing the participating financial institutions to define the degree of automation for that particular task or for the overall workflow. A task can have one or more attributes defined or associated with it, and these attributes can be associated with tasks in real-time at a process design stage (i.e., using the workflow designer 24). Examples include: whether or not a particular task is an automatic, automatic timed, automatic conditional, person-based, person-timed, timer based; ; whether a particular task involves data & code encapsulation to perform "what-if" scenarios and compute and generate automatic loan offers in seconds; as well as whether a particular task has an internal interface to another task or an external interface to another task; whether a particular task has a single document or multiple documents attached to it; whether a particular task involves single or multi bank branches; whether a particular task involves partner collaboration (meaning data inputs from their product offerings like auto prices, discounts, taxes, fees, inventory availability, etc.) as required by the loan engine to perform the logical mathematical calculations for pricing the loan offer; whether a particular task involves automatic status updates by way of internal system messages, email alerts, automatic system updates in real-time; whether a particular task at later stages needs process optimization, looping back, looping forward, or adding, inserting or deleting any task within the loan processing checklist (before or after the task at issue); whether any pricing or pricing adjustments or discounts are to be performed or automatically computed at that task; whether any specific automated "what if" analyses are to be performed; whether any PKI or digital signatures are required as part of that task approval; whether or not any role-based approvals are required by that task; whether any automatic routing is required by that task; whether any automatic reports are generated by that task; whether any automatic interface is invoked by that task; and whether any automatic analytics are generated by that task. This comprehensive functionality has been built in the designer component by creating an underlying software architecture based on logical business process rules management concepts, utilizing and underlying services oriented architecture (SOA) approach and web services, and designing the Dynamic Link Libraries (DLLs) to make available such "objects" which provide functionalities to the workflow tasks (or functions); also the under laying foundation adopts a collaborative and convergence data architecture allowing partner data collaboration (inventory management, product pricing, discounts, inventory updates, etc., which are data elements used by the workflow engine in the computation of loan offers, etc). The overall combination of these underlying architectural components allows defining the extent of automation required for that particular task of the workflow, and provides the enterprise workflow with a highly automated nature.

Also, for task (of function) in a workflow, the designer component allows generating the forms or screens dynamically using a data dictionary. This approach of designer and workflow with rules and the built-in automation described above ensures faster development and deployment of loan processing systems (and also other vertical and horizontal enterprise applications using the appropriate industry-specific data dictionary), and thereby reducing the application development costs drastically; and allowing the financial institutions or enterprises to bring in newer products and services faster in the competitive environment.

A loan director is a browser-based component overseeing the applications and the workflow, and comprises components driven by the workflow designer 24 such as components for loan fulfillment, real-time status updates, real-time analytics, performance dashboards, etc. The loan director is a back-end software component comprised of web-forms as defined and created automatically for each appropriate task in the designer (as stated above), and executables that allow financial institutions to perform back-end loan processing. The software component of the loan director provides a process-based approach to back-end loan processing after loan offers are automatically generated and presented to potential borrowers in less than about 30 seconds and accepted by the borrowers in real-time. The navigational flow of the web forms is based on the order set up in the designer for each checklist for loan fulfillment, and these forms offer the required "degree of automation" as set up with the designer component for each checklist. This includes automatically invoking of the defined internal and external interfaces by: (a) sending and receiving the data and/or performing the tasks within these interfaces as required in the checklist based on the service oriented architecture and web services calls or through built-in agents; (b) loan document compliance based document preparation systems or external software packages; and (c) other defined third party systems including upload and download interfaces to host and other systems using SOA and web services. On the bases of the defined automation features set up in the designer component, any work performed within these web-forms ensures instant automatic updates to all partners involved in the loan transaction, automated status updates to the borrowers via a gateway hub, through instant messaging, e-mails, and many other features. This loan director offers extensive supervisory control for loan processing, monitoring, routing, and offer re-negotiating, all in real-time.

The loan director generates real-time analytics (in matrix and graphical form) on loans, automatic loan computations, rules applicability and scoring decisions, qualification and eligibility criteria, valuable statistical data on productivity, borrower's results against the criteria and profiling based on the rules and criteria set up in the designer component. This information is valuable real-time information, which also enables the marketing and other business units of the participating financial institutions to analyze and create real-time marketing or sales campaigns and gain additional market share. The financial institutions thus have an ability to apply, modify, and add new business rules in the appropriate checklists with the designer in real-time without having to do any modifications to the underlying software code. In other words, business users (within the participating financial institutions) can avail of the analytics output and generate additional market share by driving the rules in real-time without having to resort to costly programming effort.

The e-loan director is browser-based and comprises front-end applications, with an embedded encapsulated component acting as an "agent" between the application and the workflow engine 20 performing instant loan processing in less than about 30 seconds, once borrower data is applied to the workflow engine 20. The e-loan director component is configured to enable instant application of borrower's information to perform "what-if analysis" (based on the pertinent rules set up in the designer component), generate real-time automatic output indicating if a potential borrower is qualified or not qualified, and, if qualified, the relevant application data (based on the processes as set up in the designer component) automatically makes a web-services or SOA call to any designated internal or external defaulters or fraud system (whether residing on the system or at remote locations) to reconfirm the borrower is a genuine, and, if qualified, the relevant application data is sent by SOA or web services to single or multiple credit bureaus for credit scoring, or to a country-specific credit rating engine component, which is offered within the designer component and available to banks/financial institutions in countries where credit bureaus are non-existent, and, if meeting the risk based offers set up criteria in the loan engine (described in greater detail below) for various borrower configurations, the pricing rules are applied, and offers are generated for presentation to the borrower immediately on submission of the application form. All these logical computational calculations and application of the rules, criteria and other complex algorithms are performed in under about 30 seconds. The embedded logical software component or "agent" between the application form and the workflow engine 20 is encapsulated in such way that performs this instant functionality in an automated and real-time fashion without any human intervention. The encapsulated software component (agent) is designed to perform all what is stated above involving complex loan processing calculations based on compliance for any kind of autos, mortgage, personal, credit cards and other loans; this component is designed and architectured to perform any complex calculations of the loans as part of the country-specific lending rules, and sustains any transaction computational load.

The bank workflow setup (a/k/a loan engine) is a browser-based component where financial institutions (e.g., banks) set up data for the instant offers for particular loan categories, sub-categories, loan types and borrower entities based on the rules set up in the designer component. Within the loan engine, financial institutions have an ability to set up as many risk-based loan offers based on their individual lending criteria, the criteria against eligibility rules, income applicability criteria, complex pricing rules & adjustments/discounts, applicable credit scores, delinquency rates, offer set up, documents required for verifications, etc. Any modification, addition or deletion of a rule in a particular checklist made in the designer component will automatically show up in the loan engine in real-time. The loan engine also provides browser access to the designated components to other partners who are involved in the loan transaction, for example, auto dealers, auto manufacturers, electronic consumer durable companies, mortgage companies, airlines, hospitality companies, retailers, brokerage firms, sales agencies, title companies, etc. The loan engine also includes a browser-based admin component for setting up partners in real-time, and establishing partner's security credentials. The browser-based loan engine also has a web-based document management component seamlessly integrated which is driven by the workflow designer 24.

Administrative tools also provide components to manage security, access roles and authorization; key performance indicator (KPI) monitoring; dashboard information displays; data dictionary, workflow and rules versioning; checklist/workflow activation and deactivation, monitoring of published checklist/workflows published on various local and distributed network environments; activation of internal and external interfaces through SOA / Web Services; Management of other types of Services and Events; Application Processing Interfaces (API's) offered by the Engine to external third-party applications based on SOA components and Web Services; Queue management console for automatic load balancing for high volume loan processing; event and error logs, and partner collaborative components, etc.

A business user of a participating financial institution who is designated as a power user can create, edit, simulate, manage and publish the loan process checklists at any time with the above automated functionalities for any loan product in any system environment without having to make any changes to software programs. Once the system 10 is deployed, the business user no longer needs to get information technology departments involved to work on the software to perform customization or modifications, which are costly options.

Also, once loan offers are generated and presented to qualified borrowers, the present system offers the key functionality in the form of options to interact with the workflow engine 20 in the event the borrower's options change, such as a change in loan amount, loan term, down payment etc. The borrowers can, in an interactive mode, "play" with the system to exercise their options, have the engine generate real-time offers, and finally accept the offer that is best suited to them. All this is done in real-time, in an automatic mode with no human intervention from the financial institution side. Furthermore, once a loan offer is accepted by a borrower, the accepted offer automatically is displayed in the centralized processing component within the loan director of the associated financial institution for further processing.

In order to configure the automated real-time loan processing system, a business user uses the workflow designer 24 to set up an appropriate data dictionary (which can be imported in an XML format into the system for any product offerings), which leads to setting up one or more checklists, then defines the degree of automation for a given checklist (as stated above), sets up rules and criteria for various loan products or borrower entities based on product domain (e.g. eligibility, scoring, product pricing & discounts for offer computations, offer rules, risk criteria, etc.), invokes both internal and external interfaces (as stated above), use real-time test data to perform test simulations of checklists and rules for a particular loan product and generate analytic results for real-time analysis, "Instant Simulated Offers" provide information in both matrix and graphical form where consumers will qualify or fail for a particular loan product offering, and also provides "deviations" which business users can optimize for setting up new offers for such categories of consumers; the simulation feature allows business users to use the system 10 continuously and generate many types of offers based on their desired marketing campaigns and test the validity of the offer presentations in real time, perform process optimization including fine tuning of processes or rules if required (e.g., add a process, delete a process, modify a process, loop a process back or forth, etc.), and finally publish the checklist(s) in real time with the embedded automation and governing rules for various loan categories and for various borrower entities in any test, development, production or live system environments. Once these checklists are published from the designer component, the checklists are automatically published in the loan engine for the data set up for instant loan offers. Within the loan engine (or bank workflow setup), the business users can now set up data against these published rules based on the desired sales or marketing needs and risk-based product offerings of the financial institutions.

A key aspect is that the business users can within the designer component add, change or delete any rules in real-time without impacting any code (i.e., underlying software) or having to make any change in databases or stored procedures or system environment, which is generally the case in the prior art. (See, e.g., specification, p. 5). Once such changes are made in the designer component, the rules are automatically published in the loan engine. The financial institutions have the ability to manage these rules in real-time and create a competitive environment for large market share. The financial institutions don't have any dependence on informational technology departments to make these results happen for them. Adding, deleting, changing a rule in real-time is a complex process and in absence of the present system to make it happen, financial institutions generally resort to making continuous changes in the software code, which is time consuming and costly. This is the one of the key aspects of the present invention, which distinguishes the present technology from the known, traditional ones. This key functionality also enables the financial institutions to have a readily available options to introduce new products in real-time, and the financial institutions don't have to have to customize their existing software to make it happen.

This platform with appropriate data-dictionary and process definitions enables financial institutions to add any new checklist with automation and rules as stated above, and thereby facilitate a single platform of repository and management of all types of loan products from stages of creation (testing and modeling) to final delivery to designated environments (published), all in real-time.

The present system offers an ability to include multiple financial institutions through individual secured gateways, and for each participating financial institution to set up, operate and manage the workflow and the subsequent processes leading to publishing of the rules in real-time. Also, within the loan engine, once the loan categories, subcategories and loan types have been populated and published (from the designer component), the designated role-based business users of each participating financial institution can select loans to administer by choosing appropriate category, subcategory and/or loan type. A financial institution can analyze and enter selection criteria values (which are values for published rules) for any selection parameter (which are sub-sets of rules) the financial institution intends to use in making a loan acceptance decision. The capability and features offered for risk based offer configurations within this component allows financial institutions domain expert to set up any complex loan offer configuration, multiple offers, add-on offers, cross product offers, discounted offers, etc. Also, if no value is entered against any of the selection criteria items that selection criteria item is not used or considered in the decision logic by the workflow engine 20.

Thus, within the automated loan system 10, financial institutions compete for borrowers in real-time. Borrowers may apply for the loan at the designated loan marketplace website, or at the financial institution's website, through financial services brokerage partner or affiliate websites, through the financial institutions branches, or through designated loan kiosks, receive multiple instant offers from different lending institutions in seconds, avail of the options to interact with the loan engine in real time if their loan parameter options change (like loan amount, down payment, loan term, etc.). If the borrower's loan parameter options change, the borrower simply modifies these parameters in an interactive mode and receives any number of offers in an automated and real-time environment based on their qualification criteria and options selected, this feature provides tremendous power to the loan offerings from the financial institutions and gives the flexibility to the borrower to compare various loan offer configurations in real-time and enabling the borrower to make a quick decision to select the offer that best suits his or her needs. On acceptance of an offer, the borrower is presented with a loan offer confirmation on-line, with instructions for presentation of the required documents for loan fulfillment. For qualified borrowers with good credit record, the entire computation of the offer applicability and offer presentation is done in a few seconds.

These concepts are illustrated in the figures and explained further in the detailed description of the invention. For instance, FIG. 6 illustrates one example of a workflow that utilizes a "home mortgage purchase" checklist 84 according to the present invention. As shown in FIG. 6, an exemplary rule (e.g., a boolean expression) requires a credit score greater than 600 as an entry condition, which dynamically determines whether or not the associated function or task of generating an instant offer 88 is performed. The parameter "600" is a stored data element, and the user's credit score is obtained via a defined task through a web Services call or SOA from user input on a loan application. The use of entry conditions allows tasks (or functions) to be performed dynamically within the loan evaluation workflow, as the workflow checklist causes tasks to be accessed only as the associated entry conditions are satisfied (i.e., evaluate to true). In that way, the system and method of the present invention can perform many non-linear processing sequences in whatever sequence is dynamically determined during the execution through the evaluation of entry conditions associated with the functions in a checklist. For a particular task whose entry conditions all evaluate to true, rule-based selection criteria within that task can then be further processed. The selection criteria can resemble the entry conditions in form (e.g., as boolean expressions). FIG. 7 is a more detailed illustration of steps for processing selection criteria associated with the "evaluate credit" function 86 of FIG. 6.

Claim Rejections - 35 U.S.C. §103(a)

Claims 1-14 and 21-27 were rejected under 35 U.S.C. §103(a) as being obvious over Bengston (U.S. Pat. No. 6,728,947) in view of Nichols et al. (U.S. Pat. No. 6,018,730) in further view of Bacon et al. (U.S. Pat. No. 6,430,538).

Amended independent claim 1 requires a workflow management system that includes a compiled program kernel containing multiple differentiated tasks defined prior to runtime setup as separate functions with the compiled program, a graphical interface having a list of geometric shapes and a workspace, each geometric shape being an abstracted object-based representation of functions within the compiled program kernel, the workspace for organizing and linking multiple geometric shapes in an ordered arrangement of objects corresponding to an order in which the multiple differentiated tasks are performed by the compiled program kernel, a database for storing the arrangement of objects as a checklist as well as for storing entry conditions that are associated with each of the multiple differentiated tasks, and a data dictionary defining discrete data elements and data relationships. According to amended independent claim 1, the contents of the data dictionary are specific to a selected industry, and the entry conditions are evaluated by the compiled program kernel with respect to each of the multiple differentiated tasks such that a particular one of the multiple differentiated tasks is performed only if all of the entry conditions associated with that particular one of the multiple differentiated tasks evaluate to true.

Amended independent claim 8 requires a workflow engine containing a plurality of discrete functions defined prior to runtime setup for performing task list processing, a workflow designer having an object-based interface for configuring task lists, and an industry-specific data dictionary defining discrete data elements and data relationships that are associated with each of the plurality of discrete functions of the workflow engine. According to amended independent claim 8, the workflow designer allows entry conditions to be defined and associated with any of the plurality of discrete functions, and each entry condition is evaluated by the workflow engine with respect to each of the plurality of discrete functions such that a particular one of the plurality of discrete functions is executed by the workflow engine only if all of the entry conditions associated with that particular one of the plurality of discrete functions evaluate to true. The workflow engine performs the discrete functions for which all associated entry conditions evaluate to true in an order determined by the ordered task list to render a financial offer decision to a remote user.

Amended independent claim 22 relates to a system for programmatically rendering a process-based decision. According to amended independent claim 22, the system requires administrative tools, a decision database, a workflow engine, a dynamic data dictionary, and a two-way messaging system. The administrative tools are configured for creating process categories and checklists associated with each process and for modifying entry conditions and selection criteria associated with discrete tasks defined prior to runtime setup and available in each checklist. The entry conditions define rules that govern whether or not each of the discrete tasks is performed. The decision database is configured for storing the process categories, the checklists, the entry conditions and the selection criteria. The workflow engine is configured for automatically processing input from a remote user and generating an instant decision based on the checklist, the entry conditions and the selection criteria associated with the checklist, and the processed input associated with the entry conditions and the selection criteria. The workflow engine is also capable of securely transmitting the instant decision to the remote user, and is capable of brokering communications between the remote user and a process administrator associated with the instant decision (as well as real-time communications with any number of other parties involved in the transaction). Also according to amended independent claim 22, the dynamic data dictionary is formatted in XML, and is configured for defining data elements and data relationships specific to a selected industry, such that the dynamic data dictionary provides a dynamic fetch and store interface with the decision database and is configured to provide, translate and modify data presentation with respect to both the remote user and the workflow engine.

Amended independent claim 25 relates to a method for workflow processing and programmatic decision-making based on object-based processes stored in memory. The method of amended claim 25 requires defining a plurality of differentiated tasks at a pre-runtime software design stage, receiving input from a remote source, determining programmatically an input type according to the received input using sets of entry conditions that are associated with each of a plurality of differentiated tasks and a data dictionary that defines data elements and data relationships used to process the entry conditions, retrieving automatically a stored process checklist from a decision database according to the input type using the data dictionary as an interface between the stored process checklist and the sets of entry conditions and as an interface between the entry conditions and both the data elements and the data relationships, processing programmatically the received information utilizing one or more of the plurality of differentiated tasks based on the entry conditions associated with the stored process checklist, rendering an automatic decision based on the processed received information, and communicating programmatically the automatic decision to the remote source. According to amended independent claim 25, each set of entry conditions is evaluated with respect to each of the plurality of differentiated tasks such that a particular one of the plurality of differentiated tasks is performed only if all of the entry conditions associated with that particular one of the plurality of differentiated tasks evaluate to true. Moreover, contents of the data dictionary are specific to a selected industry, and each entry condition of the sets of entry conditions is based upon one or more of the data elements and the data relationships defined by the data dictionary.

Bengston discloses a workflow distributing apparatus and method. The system of Bengston coordinates a serial, assembly-line style workflow that is executed by a plurality of processing devices. (Bengston, col. 4, line 66 to col. 6, line 3; FIG. 1). The system of Bengston is described with reference to coordinating printing workflows, where steps of the printing process are performed on devices at disparate locations. (E.g., Bengston, col. 1, ll. 10-47; FIG. 3). The workflow is automated so as not to require any user input while in process, and the workflow continues to until completion or until an error occurs. (Bengston, Abstract; col. 11, ll. 5-8 and 27-30; col. 14, ll. 35-39; FIG. 1). Processing by the Bengston system is decentralized, and performed sequentially by a number of different processing devices that push workflows between the processing devices in a predetermined linear fashion. (Bengston, col. 11, ll. 27-31; col. 12, line 58 to col. 13, line 8; FIG. 1). Bengston distinguishes its system from other known systems that utilize centralized processors to control the flow of information. (Bengston, col. 1, line 55 to col. 2, line 4). As noted on page 4 of the Office Action, Bengston fails to show, teach or disclose a data dictionary or entry conditions associated with tasks or functions.

Nichols et al. discloses a tutorial system installed on and run from a local workstation for helping to teach a student new skills. (Nichols, Abstract; FIG. 1). The system of Nichols et al. provides a simulated environment that students must understand and solve themselves. (Nichols et al., Abstract). Nichols et al. discloses the use of a domain model (or data dictionary) that facilitates communication of context-specific data across generic objects of an application. (E.g., Nichols et al., col. 22, ll. 18-39). The application utilizes a fixed architecture (i.e., workflow checklist) that does not include entry conditions associated with discrete functions in the checklist.

Bacon et al. discloses a workflow management system that utilizes personal subflows. Bacon et al. discloses that the personal subflows can specify rule-based branch conditions (or entry conditions), and that work flow activity is determined as a function of whether particular branch conditions evaluate to "true" or "false". (Bacon et al., col. 8, ll. 16-40; col. 9, ll. 27-38; col. 10, ll. 17-45; col. 11, ln. 46 to col. 12, ln. 60). Bacon et al. specifies that the "personal subflow does not have any explicit definition of participants or agents. Instead this information is linked or bound at run-time for the personal subflow, not at definition time." (Bacon et al., col. 3, ll. 33-36; col. 9, ll. 1-6; col. 13, ll. 1-3). In other words, workflow tasks or functions (i.e., the personal subflows) are not defined prior to runtime. Instead, Bacon et al. discloses that the personal subflows are limited constructs that are unable to associate processing activity with work items, or perform activities using more than one participant. (Bacon et al., col. 9, ll. 14-19).

None of Bengston, Nichols et al. or Bacon et al. discloses or suggests each and every limitation of amended independent claims 1, 8, 22 and 25, because none of those references discloses or suggests sets of entry conditions that are evaluated with respect to associated tasks <u>defined prior to runtime setup</u> such that a particular task in a stored process checklist is performed only if all of the entry conditions associated with that particular task evaluate to true. Both Bengston and Nichols et al. relate to workflows that utilize a pre-determined checklist (or architecture) that does not utilize entry conditions to dynamically determine the processing of the tasks during execution of the workflow. The workflows of Bengston and Nichols et al. are predefined workflows that lack the dynamic processing capabilities provided via entry conditions associated with tasks available in the workflow. Furthermore, Bacon et al. is explicit in stating that each "personal subflow does not have any explicit definition of participants or agents. Instead this information is linked or bound at run-time for the personal subflow, not at definition time." (Bacon et al., col. 3, ll. 33-36; col. 13, ll. 1-3). In this respect, the teachings of Bengston, Nichols et al. and Bacon et al. cannot be combined in the manner suggested in the Office Action.

In order to establish a *prima facie* case of obviousness, there must be some suggestion or motivation, either in the reference itself or in the knowledge generally available to one of ordinary skill in the art, to modify the reference. *In re Kotzab*, 217 F.3d 1365 (Fed. Cir. 2000); MPEP 2143.01 and 2143.03. Rejections under 35 U.S.C. §103 must also rest on a factual basis, and an examiner may not rely upon speculation, unsubstantiated assumptions or hindsight reconstruction. *In re Warner*, 37 F.2d 1011, 1017 (CCPA 1967), *cert denied*, 389 U.S. 1057 (1968). Moreover, if the proposed modification of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the cited references are not sufficient to render a claim *prima facie* obvious. *In re Ratti*, 270 F.2d 810 (CCPA 1959).

With respect to amended independent claims 1, 8, 22 and 25, the Bengston and Nichols et al. references cannot be modified to incorporate the teachings of Bacon et al. in the manner suggested in the Office Action. In particular, the <u>fixed, predefined and non-dynamic</u> workflows of Bengston and Nichols et al. cannot be modified in such a manner as to provide dynamic workflows that process tasks or functions <u>defined at runtime</u> only when associated entry conditions evaluate to true. Such a modification of the cited art would change the principle of operation of those references in a manner not taught or suggested by those references. Indeed, Bacon et al. discloses that personal subflows are defined at a runtime setup stage, using a process definition tool 105. (Bacon et al., col. 9, ll. 1-6; see also col. 3, ll. 33-36). These differences are important in determining how much of the system is defined by software developers and how much is defined by users (e.g., administrators or other designated users). The present invention, as discussed in great detail above, provides a kind of repository of defined tasks or functions that are made available to users at runtime setup when workflow checklists, entry conditions and selection criteria are created or edited, and a data dictionary can provide language that makes all of those system components meaningful for users in a selected industry. The difference between Bacon et al. and the claimed invention relates to the fact that Bacon et al. has users define both the entry conditions and the tasks or functions, whereas the present invention utilizes tasks or functions that have been defined prior to runtime setup, such that users or administrators using a workflow designer (or sub-component thereof) during runtime establish entry conditions and other workflow parameters using the previously defined tasks or functions.

Furthermore, a workflow system models a business process and allows that business process to be "run" by humans in the business environment. The business process is repeatable because it follows a model. The business process is typically associated with some set of data; thus a workflow system needs a set of data that corresponds to the model. Such data can be represented by a document, which then "flows" through the model. This document holds the business process data and the workflow system doesn't need to know anything about the nature of the data nor what the data requirements of the document are within the system. Thus, this type of document-based workflow system is more concerned with how the document is routed and who handles the document. Typically, the document is said to be in a certain "state" at each point of the business process, and that state can be tracked. The cited references all relate to this type of document-based workflow processing.

Alternatively, a workflow system can map the business process as a series of tasks necessary to perform the business process. The set of data pertaining to the business process represents the business process itself and must be known to the workflow system. A process-centric or process-based workflow system maps workflow tasks with people, transaction data and events. The present invention, as recited by independent claims 1, 8, 22 and 25, allows interaction between the workflow system and the transaction data through a data dictionary by which both the workflow engine can interact directly with transaction data and by which processes outside the workflow system can interact with the transaction data. By using a data dictionary, users of the present system and method can map tasks with transaction data dynamically. A dynamic data dictionary provides the ability to add new data dictionary fields if and when required by business users during the runtime usage of the system and/or for creating new business applications without having to make changes in the backend code (like databases/stored-procedures etc.).

Also, each process/sub-process/process instance within a workflow can be automated based on the rules (e.g., entry conditions and selection criteria) that are defined at the workflow definition level using the workflow designer, the dynamic nature of the present invention allows users to drive the rules, make modifications to the workflow, modify/add/delete rules, perform process optimization, and run workflow & rules simulations in real time. The uniqueness of the present invention as recited by independent claims 1, 8, 22 and 25 is that the users do not have to make changes in the code when any modifications the workflow is desired (which traditionally would require coding changes made by information technology department personnel), because software elements are defined prior to runtime and because of the use of a data dictionary. These recited elements allow the generation of automated workflows and applications (both vertical and horizontal) for any industry faster, rapid deployment and then have the business users to manage these without any dependence on the costly and time consuming custom programming. The cited art does not provide these features. For instance, the document-based system disclosed by Bacon et al. is a document routing system that is highly customizable but not necessarily repeatable. In other words, Bacon et al. does not allow user manipulation of workflows using tasks of function defined prior to runtime setup and using a data dictionary. Instead the system of Bacon et al. would require changes to source code made by programmers (i.e., developers). (See, e.g., Bacon et al., col. 9, ll. 27-33). Bengston and Nichols et al. also lack these elements of the present invention.

Thus, the cited references do not disclose, teach or suggest each and every limitation of amended independent claims 1, 8, 22 and 25, and the rejections of those claims under §103(a) should be withdrawn. Notification to that effect is requested.

Claims 2-7 and 21 depend from amended independent claim 1 and include all of the limitations of that base claim, claims 9-14 depend from amended independent claim 8 and include all of the limitations of that base claim, claims 23 and 24 depend from amended independent claim 22 and include all of the limitations of that base claim, and claims 26 and 27 depend from amended independent claim 25 and include all of the limitations of that base claim. Thus, for the reasons given above, dependent claims 2-7, 9-14, 21, 23, 24, 26 and 27 are likewise allowable over the cited art. The rejections of those dependent claims under §103(a) should be withdrawn, and notification to that effect is requested.

CONCLUSION

Upon review of the cited art, Applicants believe that all of the pending claims patentably define the invention over all of the art of record. Applicants believe all of the pending claims are in allowable form and respectfully request a Notice of Allowance. The Commissioner is authorized to charge payment of any additional fees associated with this paper or credit any overpayment to Deposit Account No. 11-0982.

Respectfully submitted,

KINNEY & LANGE, P.A.

Date: 3.19.2007

By: [illegible]

Austen P. Zuege, Reg. No. 57,907
THE KINNEY & LANGE BUILDING
312 South Third Street
Minneapolis, MN 55415-1002
Telephone: (612) 339-1863
Fax: (612) 339-6580